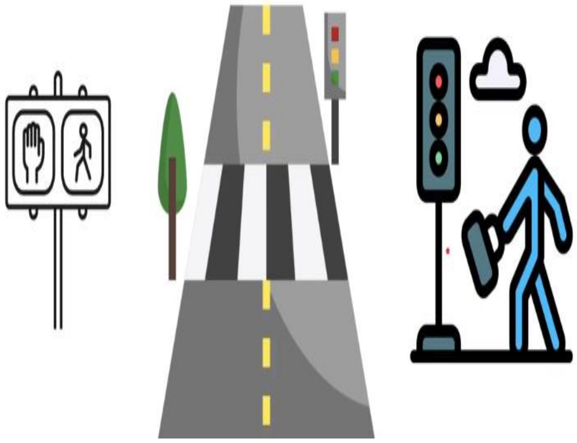


FINAL PROJECT

On-demand Traffic light control

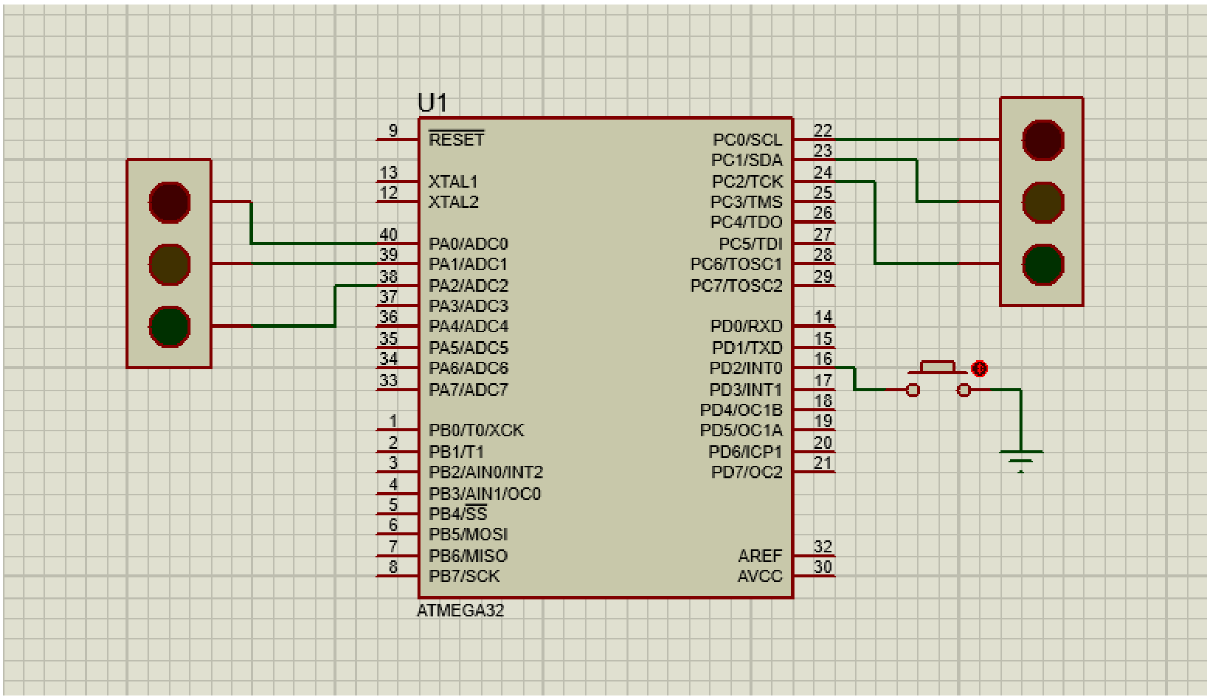
ABSTRACT

The number of vehicles in the cities are greatly increasing every day by day. This results to accidents at crossroads and due to emission of dangerous gases from exhausts as a result of over congestion of vehicles. This issue of over congestion of vehicles in one or two directions having red light while the other ends are free and having green lights forms a greater part of interest in this paper.

Prepared by: Omar Ahmed Dawood

1- <u>System Description</u>

- Introduction:
  - Traffic light control systems are widely used to monitor and control the flow of automobiles through the junction of many roads.
- Components:
  - Atmega32 Micro-Controller.
  - 2 LEDs Red, 2 LEDs Yellow, 2 LEDs Green.
  - Push Button.
- Schematic Hardware:

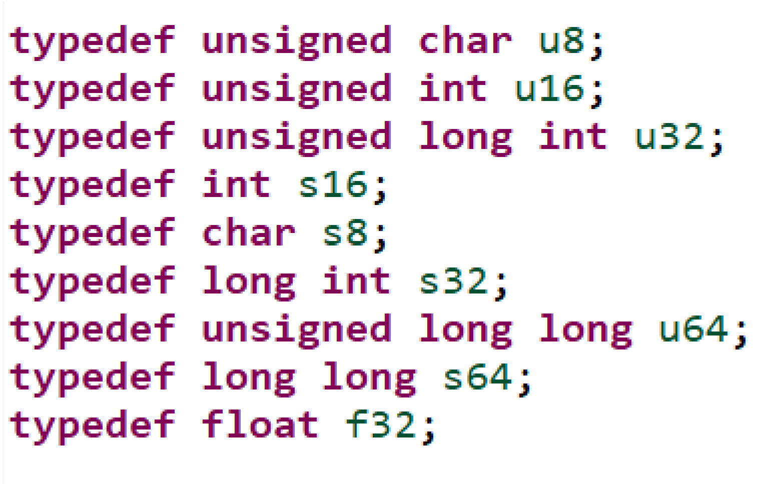


2- System Design

- System Layers:
    - Micro-Controller Abstraction Layer (MCAL) such as:
        - DIO
        - Timer
        - External Interrupt
    - App Layer Such as:
        - App.c
        - App.h
        - Main.c
- System Drivers:
    - DIO Driver
    - Timer Driver
    - External Interrupt Driver
- APIS:
    - DIO Driver:
        - DIO_voidSetDirectionPin Function
            - Determine Direction of Pin if Output or Input, arguments are Port and Pin Number and Direction.
        - DIO_voidSetValuePin Function
            - Determine State of Pin if HIGH or LOW, arguments are Port and Pin Number and State.
        - DIO_voidPullUPPin Function
            - Determine IF I want this Pin PULLUP or not, arguments are Port and Pin Number.

    - DIO_voidReadvaluePin Function
        - Get State of Pin if it was HIGH or LOW, arguments are Port and Pin Number.
- Timer Driver:
    - TIMER_voidInit Function
        - Initialization for Timer and Generate its CLK, NO Arguments.
    - Timer_voidDelayms Function
        - Delay Function that takes time delay by ms as argument.
- External Interrupt Driver:
    - INT_voidGlobalInterrupt Function
        - Enable Global Interrupt and no arguments.
    - INT0_voidInit Function
        - Initialization for peripheral Interrupt and Select interrupt mode (Falling edge) and no arguments.
    - INT0_voidEnable Function
        - Enable peripheral Interrupt and no arguments.
    - INT0_voidDisable Function
        - Disable peripheral Interrupt and no arguments.

- New Data Types That I Used:



```
typedef unsigned char u8;
typedef unsigned int u16;
typedef unsigned long int u32;
typedef int s16;
typedef char s8;
typedef long int s32;
typedef unsigned long long u64;
typedef long long s64;
typedef float f32;
```

3- System Flow

In Normal Mode:

- Cars' LEDs will be changed every five seconds starting from Green then yellow then red then yellow then Green.
- The Yellow LED will blink for five seconds before moving to Green or Red LEDs.

In Pedestrian Mode:

- Change from normal mode to pedestrian mode when the pedestrian button is pressed.
- If pressed when the cars' Red LED is on, the pedestrian's Green LED and the cars' Red LEDs will be on for five seconds, this means that pedestrians can cross the street while the pedestrian's Green LED is on.
- If pressed when the cars' Green LED is on or the cars' Yellow LED is blinking, the pedestrian Red LED will be on then both Yellow LEDs start to blink for five seconds, then the cars' Red LED and pedestrian Green LEDs are on for five seconds, this means that pedestrian must wait until the Green LED is on.
- At the end of the two states, the cars' Red LED will be off and both Yellow LEDs start blinking for 5 seconds and the pedestrian's Green LED is still on.
- After the five seconds the pedestrian Green LED will be off and both the pedestrian Red LED and the cars' Green LED will be on.
- Traffic lights signals are going to the normal mode again.

4- System Constrains

- If I made long press on the button, nothing to be done.
- If I made double press on the button, the first press will do the action and nothing to be done after the second press.